To get started on a lab computer:
1. Open a Terminal window
2. Type sqlite3 to start the sqlite3 DBMS.
3. Typing sqlite3 <filename> will attempt to open the specified database file.

Useful sqlite3 commands:
1. sqlite3 commands begin with a .
2. .help
   Lists some (not so helpful) help instructions.
3. .mode column
   .width 10
   When used together, these two commands will make the output easier to read.
4. .tables
   Gives the names of all tables contained in a database.

SQL commands:

1. General info:
   a) SQL commands are terminated by a ;
   b) SQL commands in most DBMSs are largely case-insensitive. In the example/code below, all caps are used to designate parts of the SQL syntax, while lowercase descriptions contained in <> indicate information supplied by the user which is specific to his/her database.

2. Summary of SQL commands we've learned:
   a) We can create or remove tables with the CREATE TABLE and DROP TABLE SQL statements. We can specify what fields to put in a table and what attributes they should have.
   b) We can put data (records) into a table in our database using the SQL statement INSERT.
   c) We can query our database and choose certain records using the SELECT statement.
   d) We can modify data in tables in our database using the SQL statement UPDATE.
   e) We can delete an entire record (or multiple records) from our database using the SQL statement DELETE.
   f) We can use WHERE clauses and conditions with the SELECT, INSERT, UPDATE, and DELETE statements to limit or filter what records are returned by those SQL statements.
   g) Most DBMSs have functions built into their version of SQL. These functions can perform computations on a set of records and give us the result. Examples are AVG(), SUM(), and COUNT().

3. Creating/Removing tables and putting fields in tables.
   a) General form: CREATE TABLE table1 (field1, field2...);
   b) Can add attributes to fields to instruct database on special properties.
      • NOT NULL (Means that this field cannot be empty when new records are created.)
      • PRIMARY KEY (Note that primary keys must be unique and cannot be null.)
      • Datatypes such as INTEGER, TEXT(number of characters), or DECIMAL.
      • Example:
        CREATE TABLE table1 (id INTEGER PRIMARY KEY, name text(30) NOT NULL, address TEXT(25));
   c) Removing a table (and all data contained in it) from a database:
      DROP TABLE <tablename> ;
4. How to tell how a table is constructed:
   
   \texttt{pragma table_info(tablename);}

5. The output includes the column number, name, data type for the field, and whether or not the column can be NULL (empty).

6. How to put data into our table:
   a) General form: \texttt{INSERT INTO <table name> <field list> VALUES (<data values>);}
   b) If we're inserting data for more than one field, we must separate the fields and values with commas.
   c) Example: \texttt{INSERT INTO table1 (name, address) VALUES ("Bob", "123 Win Way");}

7. Writing queries to get data from our table.
   a) Select statements give you a list of the field values from selected records in the table.
   b) General form: \texttt{SELECT <field list> FROM <tablename>;

   - * is a wildcard character. It means "everything". We can select all fields for all records using the query:
     \texttt{SELECT * FROM <tablename>;

   - If we want to select only a few fields from a table, we can use a query like:
     \texttt{SELECT field1, field2 FROM <tablename>;

   c) Usually, we want to select records from a table based on certain criteria. To do this, we use \texttt{WHERE} clauses to limit the number of records returned by our SQL statements. \texttt{WHERE} clauses contain the keyword \texttt{WHERE} followed by criteria/conditions which evaluate to true or false for each record in our table. Then our \texttt{SELECT} statement will return only those records which match the criteria we specified.
    
    - Examples:
      \texttt{SELECT * FROM table1 where id > 3;}
      \texttt{SELECT id FROM table1 WHERE name="West";}

   d) We can filter the records returned even more by using the \texttt{AND} and \texttt{OR} keywords to specify multiple criteria/conditions.
    
    - Examples:
      \texttt{SELECT * FROM table1 WHERE name="West" AND amount < 50;}
      \texttt{SELECT id, amount FROM table1 WHERE name = "West" OR amount < 40;}

8. Databases can also perform calculations for us. We can use these functions in different parts of queries to make even more powerful queries.
   a) \texttt{SUM(field)}
    
    - Unsurprisingly, this function returns the sum of the parameter it's given.
    
    - Example:
      \texttt{SELECT SUM(amount) FROM table1;}
   b) \texttt{AVG(field)}
    
    - Returns the average of the parameters it's given.
    
    - Example:
      \texttt{SELECT AVG(amount) FROM table1;}
   c) \texttt{COUNT(field)}
    
    - Returns the count of the number of items selected
    
    - Often used with conditions to verify the number of records returned.
    
    - Example:
      \texttt{SELECT COUNT(*) FROM table1 WHERE id > 3;}
      \texttt{SELECT COUNT(*) FROM table1 WHERE Name="West" OR ID=6;
9. How to update records
   a) We can update individual records in a database by using the UPDATE statement paired with a WHERE clause. For example, if an employee moves and we need to update her address, we can do this by updating the record in the table with her new information.
   b) General form: UPDATE <tablename> SET field1=value1, field2=value...;
   c) Example:
      UPDATE table1 SET address="1234 New St." WHERE id=435;
      UPDATE table1 SET address="1234 New St., age=34 WHERE id=435;
   d) If you do not use a WHERE clause or filter the records based on some criteria, ALL records will be updated with new information. For example, if we were to execute the statement:
      UPDATE table1 SET address="1234 New St";
      ALL records in table1 would have the same address of 1234 New St.

10. How to delete records:
    a) General form: DELETE FROM <tablename>;
    b) Example:
       DELETE FROM table1 WHERE id=3;
    c) As with the UPDATE statement, you need to limit the number of records returned by the statement. Executing the statement
       DELETE FROM table1;
       will remove ALL records from table1.